ĐẠI HỌC QUỐC GIA VIỆT NAM - THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP.HCM
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

NGÔN NGỮ LẬP TRÌNH NGUYÊN LÝ - CO3005

NHIỆM VỤ 3

kiểm tra tĩnh

HỒ CHÍ MINH, 01/2023

NHIỆM VỤ 3

Phiên bản 1.1

Sau khi hoàn thành nhiệm vụ này, bạn sẽ có thể

- giải thích các nguyên tắc làm thế nào trình biên dịch có thể kiểm tra một số ràng buộc ngữ nghĩa như tính tương thích kiểu, ràng buộc phạm vi,... và
- viết một chương trình Python trung bình (300 - 500 dòng mã) để thực hiện điều đó.

1 Thông số kỹ thuật

Trong bài tập này, bạn được yêu cầu viết trình kiểm tra tĩnh cho chương trình được viết bằng MT22. Để hoàn thành nhiệm vụ này, bạn cần phải:

- Đọc kỹ đặc tả ngôn ngữ MT22
- Tải về và giải nén file assignment3-initial.zip
- Nếu bạn tự tin với Bài tập 3 của mình, hãy sao chép MT22.g4 của bạn vào src/main/mt22/parser và ASTGeneration.py của bạn vào src/main/mt22/astgen và bạn có thể kiểm tra Bài tập 2 của mình bằng cách sử dụng đầu vào MT22 như lần đầu tiên tam thi (400-402).
- Mặt khác (nếu bạn chưa hoàn thành Bài tập 2 hoặc bạn không tự tin với Bài tập 3 của mình), đừng lo lắng, chỉ cần nhập AST làm đầu vào cho bài kiểm tra của bạn (như bài kiểm tra 403-405).
- Sửa đổi StaticCheck.py trong src/main/mt22/checker để triển khai trình kiểm tra tĩnh và sửa đổi CheckSuite.py trong src/test để triển khai 100 trường hợp thử nghiệm để kiểm tra mã của bạn.

2 MT22 Ngữ nghĩa

Hầu hết các mô tả ngữ nghĩa đã được đưa ra trong Đặc tả MT22. Tuy nhiên, một số trong số chúng được thảo luận sau vì sự nhầm lẫn giữa hai giai đoạn: phân tích cú pháp và kiểm tra tĩnh. Mô tả dưới đây sẽ được sử dụng trong bài tập 3 và 4.

2.1 Chuyển đổi ngầm định số nguyên và số float

Trong MT22, cho phép sử dụng số nguyên cho các phương án float nhưng ngược lại thì không. Ví dụ:

```
1 a : float = 1 ; 2
b : float = a + 2 ; 3 c :
số nguyên = 2 . 3 ;
```

Trong dòng 1, số nguyên 1 được chuyển đổi hoàn toàn thành 1.0 (ở dạng float). Ở dòng 2, số nguyên có giá trị 2 tự động chuyển thành 2.0 khi thêm vào biến float a. Tuy nhiên, phép gán ở dòng 3 không được phép.

Mọi toán tử (trong phần 6 - Đặc tả MT22) hoạt động với hai loại: số nguyên và số float đều phải tuân theo các quy tắc:

- Nếu ít nhất một toán hạng có kiểu float, kiểu kết quả sẽ là float.
- Ngược lại, kiểu kết quả của biểu thức sẽ là số nguyên.

2.2 Khai báo kiểu auto

- Một biến loại auto biểu thị một loại tự động sẽ được suy ra bởi giá trị đã cho ở phía bên tay phải.

  Ví dụ, trong đoạn mã sau, a có kiểu số nguyên, b có kiểu chuỗi và c có kiểu boolean:

  ```
  a : tự động = 1 0;
  b : auto = " xin chào "
  ; c : auto = a < 100;
  ```

  Vì vậy, việc khởi tạo khai báo biến là bắt buộc nếu kiểu là auto.

- Một hàm có kiểu trả về là auto sẽ được suy ra từ lần sử dụng đầu tiên (gọi hàm trong biểu thức hoặc trong câu lệnh gọi).

  Ví dụ:

  ```
  foo : chức năng tự động ( a : số nguyên ,      b : số nguyên ) {}
  ```

  Có ba trường hợp sử dụng:

```
1 a : float = foo ( 1 2 ) ; 2 b :
số nguyên = foo ( 1 2 ) + 1 ; 3 foo ( 1
2 ) ;      ,
```

  - Trường hợp 1: Kiểu trả về của foo sẽ được suy ra bởi kiểu của kiểu bên trái,
    trôi nổi
  - Trường hợp 2: Mỗi toán tử làm việc với cùng một loại toán hạng nên vế phải của + ở dạng nguyên dẫn đến kiểu trả về của hàm foo là ở dạng nguyên.
  - Trường hợp 3: Câu lệnh foo trong lời gọi sẽ được suy ra là một thủ tục kiểu void.

2.3 Tính năng kế thừa

Trong MT22, một hàm (được gọi là hàm con) có thể kế thừa một số tham số có thể được khai báo bằng từ khóa inherit from other function (được gọi là hàm cha). Sự kế thừa này là đơn lẻ nhưng có thứ bậc. Trong phần thân của hàm, nó có thể kích hoạt lệnh gọi hàm cha với super(<args-list>) hoặc ngăn tự động ngăn kích hoạt mặc định bằng cách gọi preventDefault(). Các quy tắc này như sau:

- Nếu hàm cha có danh sách tham số không rỗng, thì câu lệnh đầu tiên trong hàm con phải là super(<args-list>) với danh sách tham số phù hợp để gọi hàm cha hoặc preventDefault() để ngăn kích hoạt.
- Mặt khác (một danh sách tham số trống), việc kích hoạt chức năng cha mẹ là ẩn. Để dừng việc gọi tự động này, lệnh đầu tiên trong hàm con phải là preventDefault().

Với các tham số có từ khóa inherit thì hàm con sẽ kế thừa chúng trong phạm vi của nó nên việc khai báo lại sẽ không hợp lệ.

3 Trình kiểm tra tĩnh

Trình kiểm tra tĩnh đóng một vai trò quan trọng trong các trình biên dịch hiện đại. Nó kiểm tra trong thời gian biên dịch xem chương trình có tuân thủ các ràng buộc ngữ nghĩa theo đặc tả ngôn ngữ hay không. Trong nhiệm vụ này, bạn được yêu cầu triển khai trình kiểm tra tĩnh cho ngôn ngữ MT22 phạm vi tĩnh.

Đầu vào của trình kiểm tra nằm trong AST của chương trình MT22, tức là đầu ra của phép gán 2.
Đầu ra của bộ kiểm tra không có gì nếu đầu vào được kiểm tra là chính xác, nếu không, một thông báo lỗi sẽ được đưa ra và bộ kiểm tra tĩnh sẽ dừng ngay lập tức.

Đối với mỗi lỗi ngữ nghĩa, sinh viên nên đưa ra ngoại lệ tương ứng được đưa ra trong StaticError.py bên trong thư mục src/main/mt22/checker/ để đảm bảo rằng nó sẽ được in ra giống như mong đợi. Mỗi testcase có nhiều nhất một loại lỗi. Các ràng buộc ngữ nghĩa cần thiết để kiểm tra nhiệm vụ này như sau.

3.1 Khai báo lại Biến/Tham số/Hàm

Tuyên bố phải là duy nhất trong phạm vi của nó. Mặt khác, ngoại lệ Redeclared(<kind>, <identifier>) được giải phóng, trong đó <kind> là loại (Biến/Tham số/Hàm) của mã định danh trong khai báo thứ hai.

3.2 Mã định danh/Chức năng không được khai báo

- Ngoại lệ Undeclared(Identifier(), <identifier-name>) được giải phóng khi có một số nhận dạng được sử dụng nhưng không thể tìm thấy khai báo của nó. Định danh có thể là một biến

hoặc tham số.

- Undeclared(Function(), <function-name>) được giải phóng nếu không tồn tại bất kỳ chức năng nào có tên đó. Việc sử dụng chức năng (để kế thừa hoặc gọi) có thể được cho phép trước khi khai báo.

3.3 Khai báo biến/tham số không hợp lệ

- Ngoại lệ Invalid(Biến(), <tên-biến>) được giải phóng khi một biến được khai báo theo kiểu auto mà không cần khởi tạo.
- Ngoại lệ Invalid(Parameter(), <parameter-name>) được giải phóng khi một tham số ter được khai báo lại trong một hàm với hàm cha có cùng tham số trong tên với từ khóa inherit.

3.4 Nhập không khớp trong biểu thức

Một biểu thức phải tuân thủ các quy tắc loại cho biểu thức, nếu không, ngoại lệ TypeMis matchInExpression(<expression>) sẽ được giải phóng. Các quy tắc loại biểu thức như sau:

- Đối với một mảng subscripting (toán tử chỉ mục) E1[E2], E1 phải ở kiểu mảng và E2 phải là một danh sách các số nguyên.
- Đối với biểu thức nhị phân và đơn nguyên, các quy tắc loại được mô tả trong đặc tả MT22 và chuyển đổi ngầm định trong tiểu mục 2.1.
- Đối với một lệnh gọi hàm <tên hàm>(<args>), callee <tên phương thức> phải có kiểu trả về là non-void. Các quy tắc loại cho các đối số và tham số giống như các quy tắc được đề cập trong lời gọi thủ tục.

3.5 Loại không khớp trong tuyên bố

Một câu lệnh phải tuân thủ các quy tắc loại tương ứng cho các câu lệnh, nếu không, ngoại lệ TypeMismatchInStatement(<câu lệnh>) sẽ được giải phóng. Các quy tắc loại cho các câu lệnh như sau:

- Loại biểu thức điều kiện trong câu lệnh if/while/do-while phải là boolean.
- Trong câu lệnh for, kiểu biến vô hướng, biểu thức cập nhật phải là số nguyên.
- Đối với câu lệnh gán, vế trái có thể ở bất kỳ kiểu nào trừ kiểu void và kiểu mảng. Phía bên tay phải (RHS) cùng loại với LHS hoặc thuộc loại có thể ép buộc với loại LHS.

- Đối với việc truyền tham số, quy tắc gán được áp dụng cho việc truyền tham số trong đó tham số được coi là LHS và đối số tương ứng là RHS.
- Đối với một câu lệnh trả về, biểu thức trả về có thể được coi là RHS của một phép gán ngầm mà LHS là kiểu trả về.

3.6 Ngắt/Tiếp tục không trong vòng lặp

Câu lệnh ngắt/tiếp tục phải trực tiếp hoặc gián tiếp bên trong vòng lặp, nếu không thì phải ném ngoại lệ MustInLoop(<câu lệnh>).

3.7 Mảng không hợp lệ

Phải ném ngoại lệ IllegalArrayLiteral(<array literal>) trừ khi tất cả các ký tự trong một ký tự mảng phải cùng loại.

Ví dụ: {1, 2.0} là một ví dụ về lỗi này.

3.8 Tuyên bố đầu tiên không hợp lệ

Trong các hàm kế thừa, câu lệnh đầu tiên tuân theo quy tắc trong tiểu mục 2.3, nếu không, ngoại lệ UnlimitedStatementInFunction(<tên-hàm>) sẽ được giải phóng.

3.9 Không có điểm đầu vào

Phải có một hàm có tên là main không có bất kỳ tham số nào và trả về kiểu void trong chương trình MT22. Nếu không, ngoại lệ NoEntryPoint() sẽ được giải phóng.

4 đệ trình

Bài tập này yêu cầu bạn gửi 2 tệp: StaticCheck.py chứa lớp StaticChecker với kiểm tra phương thức nhập và CheckSuite.py chứa 100 trường hợp kiểm tra.

Tệp StaticCheck.py và CheckSuite.py phải được gửi trong "Bài tập 3 - Gửi".

Hạn chót được thông báo trên trang web của khóa học và đó cũng là nơi bạn PHẢI gửi mã của mình.

5 đạo văn

Bạn phải tự mình hoàn thành nhiệm vụ và đừng để người khác nhìn thấy công việc của bạn.

Nếu bạn vi phạm bất kỳ yêu cầu nào, bạn sẽ bị trừng phạt theo quy định của trường đại học vì tội đạo văn.

6 Nhật ký thay đổi

- Loại void với câu lệnh gọi.